Here is the main.c file for the KL43Z

```
/*
 * Copyright 2016-2018 NXP Semiconductor, Inc.
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
modification,
 * are permitted provided that the following conditions are met:
 *
 * o Redistributions of source code must retain the above copyright
notice, this list
 *   of conditions and the following disclaimer.
 *
 * o Redistributions in binary form must reproduce the above copyright
notice, this
 *   list of conditions and the following disclaimer in the
documentation and/or
 *   other materials provided with the distribution.
 *
 * o Neither the name of NXP Semiconductor, Inc. nor the names of its
 *   contributors may be used to endorse or promote products derived
from this
 *   software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
"AS IS" AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED
 * WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
ARE
 * DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS
BE LIABLE FOR
 * ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES
 * (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES;
 * LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
CAUSED AND ON
 * ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR
TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE
USE OF THIS
 * SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

```
 */

/**
 * @file    MKL43Z256xxx4_Project_Final.c
 * @brief   Application entry point.
 */
#include <stdio.h>
#include "board.h"
#include "peripherals.h"
#include "pin_mux.h"
#include "clock_config.h"
#include "MKL43Z4.h"
#include "fsl_debug_console.h"
#include "hcc_terminal.h"
#include "fsl_port.h"

/*! @name PORTD3 (number 25)
  @{ */
#ifndef BOARD_TEST_GPIO
#define BOARD_TEST_GPIO GPIOD /*PORT GPIO PORT: Port D*/
#endif
#define BOARD_TEST_PORT PORTD /*!<@brief PORT device name: PORTD */
#ifndef BOARD_TEST_PIN
#define BOARD_TEST_PIN 3U     /*!<@brief PORTD pin index: 3 */
#endif                                  /* @} */


void BOARD_InitMyPins(void) {
     gpio_pin_config_t TEST_config = {
             .pinDirection = kGPIO_DigitalOutput,
             .outputLogic = 1U
         };
      /* Initialize GPIO functionality on pin PTD5 (pin 6)  */
         GPIO_PinInit(BOARD_TEST_GPIO, BOARD_TEST_PIN, &TEST_config);

         const port_pin_config_t TEST = {
          /* Internal pull-up/down resistor is disabled */
          kPORT_PullDisable,
          /* Slow slew rate is configured */
          kPORT_SlowSlewRate,
          /* Passive filter is disabled */
          kPORT_PassiveFilterDisable,
          /* Low drive strength is configured */
          kPORT_LowDriveStrength,
          /* Pin is configured as PTD5 */
          kPORT_MuxAsGpio};
```

//This File must be included, more on that later

These Initializations must be done for any pin used (can be named however you want)

Created Function

Config. as output

Allows for changing of different parameters (Pull-up, Drive Strength, etc.)

```
            /* PORTD5 (pin 6) is configured as PTD5 */
            PORT_SetPinConfig(BOARD_TEST_PORT, BOARD_TEST_PIN, &TEST);
}

//LED FUNCTION
static void cmd_led(char *param)
{
      //Flips the state of the LED's
  static led1_on=0;
  static led2_on=0;
  int i;

  while((*param!='1')&&(*param!='2')&&(i<10)) {//Makes sure that an
LED is chosen
        param++;
        i++;
  }

  if (*param=='1'){
        if (led1_on)
        {
              led1_on=0;
              GPIO_PinWrite (GPIOD, 3, 1);
              //LED_RED_ON();
        }
        else
        {
              led1_on=1;
              GPIO_PinWrite (GPIOD, 3, 0);
              //LED_RED_OFF();
        }
  }
  if (*param=='2'){
        if (led2_on)
        {
              led2_on=0;
             LED_GREEN_ON();
        }
        else
        {
              led2_on=1;
             LED_GREEN_OFF();
        }
   }
  *param='0';
}
```

Write to pins (GPIO_WritePinOutput actually just calls this function, so this is cleaner)

```
static const command_t led_cmd = {
  "led", cmd_led, "   Toggles leds state"
};
/*
 * @brief   Application entry point.
 */
//TEST_PIN_INIT(1);// 11/15 RM

int main(void) {
      char c;
      /* Init board hardware. */
    BOARD_InitBootPins();
    BOARD_InitBootClocks();
    BOARD_InitBootPeripherals();
      /* Init FSL debug console. */
    BOARD_InitDebugConsole();
    BOARD_InitLEDs(); //THIS FUNCTION ALSO SETS UP THE CLOCK FOR PORTS
D AND E
    BOARD_InitMyPins();
    PRINTF("Hello World\n");
    //Modified to work with KL43Z
    terminal_init(PUTCHAR, GETCHAR, PUTCHAR);
    (void)terminal_add_cmd((command_t*)&led_cmd);
    /* Force the counter to be placed into memory
    volatile static int i = 0 ;
    /* Enter an infinite loop, just incrementing
    while(1) {

        i++ ;
        terminal_process();
    }
    return 0 ;
}
```

Function is called in ‘main’ here

Notice also that the Init LED’s is highlighted. This sets up the clock for ports D and E. If other ports are desired, these clocks must be set up (These may already be done in other functions above, I haven’t checked)